

PROSIDING

Seminar Nasional Informatika 2005

SNI 2005



PROSIDING

KONTRIBUSI PENDIDIKAN TINGGI

DALAM PENGEMBANGAN INDUSTRI MOBILE COMPUTING

AUDITORIUM KAMPUS III UNIVERSITAS AHMAD DAHLAN, YOGYAKARTA, INDONESIA

SABTU, 27 AGUSTUS 2005



Program Studi Teknik Informatika
Universitas Ahmad Dahlan



75

PROSIDING

Seminar Nasional Informatika (SNI) 2005
Universitas Ahmad Dahlan Yogyakarta

Kontribusi Perguruan Tinggi dalam
Pengembangan *Mobile Computing*

Auditorium Kampus III Universitas Ahmad Dahlan Yogyakarta
27 Agustus 2005

Editor :

Ardiansyah, S.T.
Eko Aribowo, S.T., M.Kom
Hasanuddin, S.T.

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN

Prosiding Seminar Nasional Informatika (SNI) 2005

Kontribusi Perguruan Tinggi dalam Pengembangan *Mobile Computing*

Sabtu, 27 Agustus 2005, Auditorium Kampus III Universitas Ahmad
Dahlan Yogyakarta

Hak Cipta © 2005 pada Penulis

ISBN : 979.756-077-7

Hak Publikasi pada Program Studi Teknik Informatika Fakultas
Teknologi Industri Universitas Ahmad Dahlan Yogyakarta.

Dilarang memperbanyak sebagian atau seluruh isi dari buku ini dalam
bentuk apapun, tanpa ijin tertulis dari penerbit dan penulis.

PANITIA PELAKSANA
SEMINAR NASIONAL INFORMATIKA (SNI) 2005
UNIVERSITAS AHMAD DAHLAN

Advisory Committee :

Ir. Lukito Edi Nugroho, M.Sc., Ph.D (UGM)
Drs. Agus Harjoko, M.Sc., Ph.D (UGM)
Jazi Eko Istiyanto, Ph.D (UGM)
Prof. Dr. Benyamin Kusumoputro (UI)
Dr. M. Sukrisno Mardiyanto (ITB)

Organizing Committee :

Chairman : Ratna Wardani, S.Si., MT.
Members : Rusydi Umar, S.T., M.T.
Hasanuddin, S.T.
Sri Winiarti, S.T.
Dewi Soyusiawaty, S.T., MT
Eko Aribowo, S.T., M.Kom
Ardiansyah, S.T.
Murinto, S.Si., M.Kom.
Ali Tarmuji, S.T.
Muhammad Aziz, S.T.
Drs. Wahyu Pujiyono, M.Kom
Sri Handayaningsih, S.T.
M. Idham Ananta Timur, S.T., M.Kom
Taufiq Ismail, S.T.
Ir. Ardi Pujiyanta, M.T.
Drs. Tedi Setiadi, M.T.
Farida Sulistiyorini, S.T.

KATA PENGANTAR

KETUA PANITIA SNI 2005

Assalamualaikum Wr. Wb.

Salam sejahtera bagi kita semua,

Para peserta seminar yang berbahagia, pertama kali kita panjatkan puji syukur kehadirat Allah SWT yang senantiasa memberikan rahmat dan hidayah-Nya kepada kita semua sehingga pada hari ini kita dapat menghadiri Seminar Nasional Informatika (SNI) 2005 dengan tema “Kontribusi Perguruan Tinggi dalam Pengembangan Industri *Mobile Computing*”.

Mobile Computing merupakan paradigma komputasi yang memungkinkan entitas yang terlibat dan proses komputasi tidak dibatasi oleh lokasi. Sedikit berbeda dengan komputasi terdistribusi, *mobile computing* menekankan pada transparansi lokasi. Mobile computing menjadi topik yang sangat menarik untuk dikaji lebih jauh, terutama perkembangan teknologi pada industri mobile device.

Dengan seminar ini baik dalam bentuk makalah utama yang dipresentasikan oleh Key note Speech dari PT. Sun Microsystems Indonesia yaitu oleh Bapak Harry Kaligis (General Manager PT. Sun Microsystems Indonesia) maupun presentasi makalah dari para pemakalah dalam *Call For Paper* (CFP), diharapkan dapat memberikan wawasan dan pengetahuan yang sangat bermanfaat bagi kita semua dalam mengikuti perkembangan teknologi terutama teknologi mobile.

Wassalamualaikum Wr. Wb.

Ratna Wardani, S.Si., MT.

DAFTAR ISI

Kata Pengantar	iv
Daftar isi	v
Keynote Speaker Harry Kaligis GM Bussiness Development PT. Sun Miersystem Indonesia.....	viii
Drs. Wahyu Pujiyono, M.Kom, M. Idham AT, S.T, M.Kom, Ade Irma Suryani <i>Membangun Aplikasi Electronic Govermernt Pelayanan Administrasi Pajak</i> <i>Kendaraan Bermotor Berbnasis Web.....</i>	1
Rimba Whidiana Ciptasari, S.Si, M.T <i>Digital Watermarking pada Dokumen Citra Teks</i>	8
Raymond Kosala <i>Web Mining Application for Mobile Computing</i>	15
Eko Aribowo <i>Tinjauan Umum Metode Stegnography, Watermarking dan Cryptography pada</i> <i>Pengamanan Informasi</i>	21
Suryo Hadi, I Gede Susrama <i>Sistem Penterjemah Kalimat Bahasa Inggris pada Handphone dengan J2ME</i>	27
Wahyu Pujiyono, Rusydi Umar, Kusatanti Arum Sari <i>Sistem Pelayanan Pemesanan Tiket Pesawat Via SMS</i>	39
Eko Aribowo <i>Sistem Monitoring Aktivitas dan Beban Kerja Teller (Studi kasus di Kantor Cabang Bang Internasional</i> <i>Indonesia Semarang)</i>	45
Poltak Jafferson, Vera Suryani, Achmad Rizal <i>Sistem Keamanan Rumah Multi-Sensor Terintegrasi Berbasis PC</i>	51
Ardi Pujiyanta, Tedy Setiadi <i>Sistem Infomasi Geografi Untuk Mengetahui Daerah Yang Terkena Penyakit Demam Berdarah di Daerah</i> <i>Istimewa Yogyakarta</i>	55
Ardi Pujiyanta, Rusydi Umar, Budiono <i>Sistem Infomasi Geografi Obyek Pariwisata di Yogyakarta</i>	64
Subur Anugrah, Setyo Nugroho <i>Sistem Informasi Akademik Mahasiswa berbasis Web pada STMIK Balikpapan</i>	74
Murinto <i>Perbaikan Kualitas citra Menggunakan Histogram Linear Contrast Stretching pada Citra Skala</i> <i>Keabuan</i>	80
Ardi Pujiyanta, Rusydi Umar <i>Prediksi Tingkat Kesuburan Hewan Ternak dengan Jaringan Saraf Tiruan</i>	85
Tedy Setiadi, Ratna Wardani, Ismail Mahifa <i>Perangkat Lunak Bantu Komputerisasi dan Visualisasi Proses Konstruksi Retaining Wall</i>	94
Beby H. A. Manafe <i>Penggunaan Algoritma Heuristic untuk Traffic Grooming Pada Jaringan Ring SDH/WDM searah</i>	98
Muhammad Sais Hasibuan <i>Perancangan dan Implementasi Cyber Campus dengan Menggunakan Hot Spot pada Area Darmajaya</i>	108
Suryo Hadi, I Gede Susrama <i>Peningkatan Mutu Citra dengan Menggunakan Metode Sharp-Retnex</i>	113

Baharuddin <i>Peningkatan Kualitas Transmisi Citra dengan Menggunakan Teknik Diversity Combining Metode Selection Diversity</i>	121
Enny Itje Sela <i>Penggunaan Koefisien Kontingensi pada Data Mining untuk Mengetahui Faktor-faktor yang Mempengaruhi kesuksesan Alumni Perguruan Tinggi</i>	127
Muhmmad Idham Ananta Timur <i>Perbandingan Kinerja Client dalam Interoperabilitas Enterprise JavaBeans dengan Common Request Broker Architecture (COBRA)</i>	131
Dade Nurjanah <i>Pemodela Siswa Berbasis kasus pada Intelligent e-Learning System</i>	135
Tedy Setiadi, Eko Mursito, I Nyoman Kertajaya <i>Pemilihan Data MultiList Representasi Pointer untuk Pengembangan Program Pengolahan Sparse Matriks</i>	140
Yulia Rachmawati K. <i>Pemanfaatan Konsep Inheritance untuk Penanganan pesan berbasis teks</i>	145
Arif Sulisty, Desi Purwanti <i>Pemanfaatan Infrastruktur wi-fi pada mobile PC dan PDA untuk Proses E-Learning di era Intratet dan Internet</i>	152
Dedi Trisnawarman, Pung Setiadi Wijana <i>Model Sistem Penunjang Keputusan Peramalan Produksi</i>	156
Mursid W. Hananto <i>Mobile Database dengan Pocket PC</i>	163
Yoanes Bandung, Armein Z. R. Langi, Suhono H., Supangkat, Charmadi Machbub <i>Metoda IntServ dan DiffServ untuk Jaminan Kualitas Layanan di Rural Next Generation Network (R-NGN)</i>	168
Arief Rachman, S.Kom <i>Mempersiapkan E-Learning Sebagai Tuntutan Globalisasi</i>	175
Drs. Wahyu Pujiyono, M.Kom, Rusydi Umar, S.T, M.T <i>Membangun Portal E-Government Pengurusan Dokumen Ekspor secara Online dengan Pendekatan G2B (Government to Bussiness)</i>	178
Bambang Sugiyantoro <i>Membangun Aplikasi Komunikasi Berbasis Text Menggunakan Teknologi Windows Sockets (Winsock) Dan User Datagram Protocol (UDP)</i>	184
Ardi Pujiyanta, Wahyu Pujiyono, Fradika Indrawan <i>Deteksi Golongan Darah Manusia Menggunakan Metode Backpropagation</i>	194
Dwi Sulisworo, Dewi Soyusiawati, Ani Windarti <i>Layanan KRS Online Universitas Ahmad Dahlan dengan Sistem Manajemen Mutu ISO Berbasis Web</i>	201
Handoko, Saptadi Nugroho, Buyung Halim <i>Jawa Markup Language (JawaML)</i>	209
Ali Tarmuji <i>Invasive Software Composition, Paradigma Alternatif dalam Mengkomposisi. Komponen Software</i>	216
Imam Azhari <i>Implementasi Penyelesaian Travelling Salesperson Problem dengan Menggunakan Algoritma Cheapest-Insertion Heuristic</i>	222
Wahyu Pujiyono, Kharisma <i>Dialog Interaktif berbasis Web Menggunakan VoiceXML</i>	229

Eko Aribowo, Wahyu Pujiyono, Joko Purnomo Sunu <i>Desain dan Implementasi Sistem Informasi Transaksi Ekspor dan Impor dengan LC (Studi Kasus di Bank Internasional Indonesia Kancab Jogjakarta)</i>	237
Tri Kuntoro Priyambodo <i>Database Multi Resolusi pada Data Spasial Tredistribusi</i>	243
Muhammad Said Hasibuan, Zainal A. Hasibuan <i>Computer – Mediated Learning Evaluasi dan Solusi Studi kasus Scele Universitas Indonesia</i>	247
Susany Soplanit <i>Chaos Based Stream Cipher</i>	252
Murinto <i>Bilateral Filtering untuk Multi-scale Deteksi Tepi</i>	256
Cuks Subiyantoro, Buafit Nugroho <i>Aplikasi Web untuk Pengelolaan WebHosting</i>	260
Sri Winiarti, Tedy Setiadi, Peronisa Humaira <i>Aplikasi Sistem Pakar untuk membantu Diagnosa Penyakit menular Seks</i>	270
Istiadi <i>Aplikasi Sistem Pakar menggunakan media WAP dengan metode Statik</i>	280
Arif Sulistiyo <i>Aplikasi Pencarian Derah Rumah menggunakan PDA dan Mobile PC Studi Kasus pada Perumahan BSB Semarang</i>	285
Supriyono, Ichwan Taqwa <i>Aplikasi Metode Bairstow dan Metode Fadeev-Leverrier sebagai Penyelesaian Nilai Eigen untuk Sistem yang Berukuran Besar</i>	288
Arief Rachman, S.Kom, Abdul Rouf, S.Kom <i>Aplikasi Messenger Untuk Jaringan Lokal dengan Java</i>	295
Sri Winiarti, Rika Muliani <i>Aplikasi Expert System Berbasis Web untuk Mediagnosa Gangguan Produksi Sel Darah Merah</i>	298
Supriyono <i>Analisis Stabilitas Skena Eksplisit pada Penyelesaian Persamaan Konduksi Panas Dua Dimensi</i>	305
Johanes F. M. Bowakh <i>Algoritma Disain Topologi Logik Jaringan WDM Menggunakan QoR</i>	310
Jazi Eko Istiyanto, Eko Purwadi <i>Alat Pemantau Suhu Jarak Jauh berbasis SMS (An SMS-based Remote Temperatur Monitoring Device)</i>	317

Pemilihan Struktur Data MultiList Representasi Pointer untuk Pengembangan Program Pengolahan *Sparse* Matriks

Tedy Setiadi¹, Eko Mursito², I Nyoman Kertajaya³

¹Jurusan Teknik Informatika, Universitas Ahmad Dahlan

²Jurusan Teknik Fisika Institut Teknologi Bandung

³Jurusan Matematika Universitas Udayana

¹email : tedz68@yahoo.com

Abstrak

Pada pengembangan program pengelolaan matriks tentu kebanyakan pemrogram memilih menggunakan struktur data *matrix* atau array dua dimensi mengingat secara natural sesuai dengan karakteristiknya selain kemudahan dalam pengembangan algoritmanya. Namun bila matriks yang dikelola adalah *sparse* matriks maka pemilihan struktur data ini kurang cocok sebab akan terjadi pemborosan memori. Makalah ini menjelaskan pengembangan program pengelolaan *sparse* matriks,

Kata kunci: multi list, pointer, *sparse* matriks

1. Pendahuluan

1.1. Matriks

Matriks adalah susunan empat persegi panjang dari elemen-elemennya yang dinotasikan dengan $A_{m \times n}$. Matriks A di atas mempunyai m baris dan n kolom maka A disebut matriks dengan orde $m \times n$. Adapun beberapa istilah yang dikenal pada matriks, antara lain :

- matriks bujursangkar yaitu matriks dengan banyaknya baris dan kolom sama
- matriks nol yaitu matriks dengan semua elemennya bernilai nol
- matriks identitas yaitu matriks yang semua elemen diagonalnya bernilai 1 sedangkan elemen lainnya bernilai 0.

Sparse matriks (matriks jarang) yaitu matriks dimana sebagian besar elemennya bernilai nol, dengan catatan matriks ini umumnya berukuran besar mencapai ratusan bahkan ribuan baris dan kolom. [1].

Matriks dapat dikenakan operasi-operasi antara lain :
Penjumlahan : Jika A dan B matriks orde $m \times n$, maka $A + B$ adalah matriks $m \times n$ dimana $(A + B)_{ij} = A_{ij} + B_{ij}$ (dibaca "elemen ke-ij dari matriks A + B adalah berisi penjumlahan elemen ke-ij A dengan elemen ke-ij B")

Perkalian skalar : Jika A matriks $m \times n$ dan c suatu skalar, maka cA adalah matriks $m \times n$ yang berisi perkalian c dengan setiap elemen pada A; yaitu $(cA)_{ij} = cA_{ij}$

dengan menggunakan struktur data multilist representasi pointer. Dengan pemilihan struktur data ini maka terjadi penghematan memori yang cukup signifikan, karena hanya elemen yang tidak kosong yang akan dikelola dan disimpan dalam memori. Konsekuensi yang muncul dari penghematan memori yang terjadi adalah harus dibayar dengan semakin rumitnya algoritma yang harus dikembangkan. Program diimplementasikan dengan bahasa C dan mampu menangani operasi penjumlahan, pengurangan, serta perkalian dua buah *sparse* matriks.

3. Perkalian : Jika A matriks $m \times n$ dan B matriks $n \times p$, maka hasil kali A dengan B dinotasikan AB adalah matriks $m \times p$ dengan elemen ke $_{ij}$
 $(AB)_{ij} = A_{i1} B_{1j} + A_{i2} B_{2j} + \dots + A_{in} B_{nj} =$
dengan $1 \leq m, 1 \leq p$

1.2. Aplikasi *Sparse* Matriks

Sparse matriks banyak dijumpai dalam aplikasi sains dan teknik. Misal, aplikasi pada rekayasa sipil, model persamaan matematika pada tiang jembatan besar. kebanyakan koefisien pada sembarang persamaan adalah nol. Pemikiran yang sama terjadi pada aplikasi model bangunan dan struktur fisik lainnya.

Pada rangkaian listrik, model dari rangkaian tersebut juga *sparse* semakin besar rangkaiannya maka semakin besar sparsitasnya. Jadi, meskipun prinsipnya dapat dihubungkan dua titik jauh pada rangkaian listrik dengan kabel, dalam kebanyakan rangkaian ini sangat kecil dilakukan.

1.3. Struktur Data

Struktur data yang dikenal dalam pemrograman adalah struktur data dengan tipe :

1. Dasar : integer, real, boolean, dan character, nama informasi yang didefinisikan setiap saat hanya dapat menyimpan satu nilai
2. Bentukun, yaitu tipe yang merupakan komposisi dari tipe dasar, nama informasi yang didefinisikan setiap saat hanya mengandung satu nilai sesuai dengan komposisi dari tipe yang didefinisikan

3. Tabel (array), yaitu tipe yang mendefinisikan sekumpulan elemen tabel bertipe sama, dan nantinya akses elemen tersebut akan dialokasikan secara kontigu, dengan akses elemen yang ditentukan oleh ondeks. Nama informasi yang didefinisikan setiap saat dapat menyimpan banyak nilai, tergantung pada ukuran tabel.

Ketiga tipe tersebut sesuai dalam penyimpanannya dalam memori, dan tidak ada lagi pilihan untuk menentukan implementasinya.

Dalam penulisan program prosedural adalah menentukan strukturisasi penyimpanan informasi (kamus) dan aksi yang memanipulasi struktur tersebut [3]. Jika tempat penyimpan nilai pada nama informasi dianalogikan sebagai sel maka isi dari sel dapat diacu dari letak atau alamat sel tersebut di memori. Karena dalam pemrograman, data yang diproses seringkali sangat kompleks, sedangkan struktur dalam penyimpanan komputer sangat sederhana maka dibutuhkan suatu abstraksi dari penyajian data yang kompleks. Adapun struktur data abstrak yang standar dibidang informatika adalah :

- List linier
- Multi List
- Stack (Tumpukan)
- Queue (Antrian)
- Tree (Pohon)
- Graph (Graf)

1.4. Struktur Data Penyimpanan Sparse Matriks

Misal dipunyai matriks A berorde 10 x 10. Dari seluruh 100 elemen, hanya ada 11 elemen tidak nol. Untuk menyimpan sparse matriks seperti di atas tidak pernah dilakukan dengan menyimpan semua elemennya termasuk elemen nol, karena akan membutuhkan ruang yang besar yang tidak efisien.

Ada beberapa struktur data yang dapat digunakan untuk menyimpan elemen tidak nolnya saja dari sparse matriks, yang pada prinsipnya adalah dengan struktur data kontigu yaitu disimpan secara berurutan atau menggunakan struktur data linked-list [2].

1. Struktur Tabel (Array 2 dimensi)

Deklarasi nya

Type Matriks = array (1..10, 1..10) of real

Cara ini adalah cara paling mudah tetapi harus dibayar dengan pengorbanan yang sangat mahal dalam hal penyimpanannya karena semua elemen nolnya juga akan ikut tersimpan. Hal ini akan sangat memboroskan memori, dengan alasan inilah maka penyimpanan *sparse* matriks menggunakan deklarasi ini tidak pernah dilakukan.

2. Struktur Array

Dengan menyimpan elemen yang tidak nol saja, menggunakan array satu dimensi (vektor). Dalam hal ini

diperlukan 3 buah vektor, yang masing-masing digunakan untuk menyimpan nomor baris, nomor kolom dan nilai elemen yang tidak nol.

Pendeklarasian struktur datanya sebagai berikut :

B_Nilai : (terdefinisi, banyaknya elemen tak nol)

Type I_Matriks = record

No_Baris : integer;
No-Kolom : integer;
Nilai : real;
End of record

Type T_SparMat = array [1..B_Nilai] of I_Matriks

Indeks	Baris	Kolom	Nilai
1	1	4	2
2	1	8	3
3	2	1	5
4	4	2	1
5	5	10	7
6	6	3	-2
7	6	4	-5
8	8	9	8
9	8	10	-1
10	9	8	4
11	10	6	-17
12	10	7	9

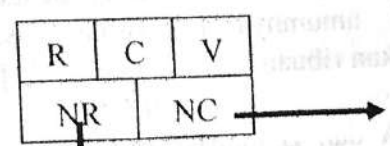
Dengan

memperhatikan gambar tersebut, maka cara penyimpanan *sparse* matriks masih boros, karena jika pada suatu baris mempunyai lebih dari satu elemen tak nol, maka nomor baris tersebut disimpan beberapa kali.

2. Perancangan Struktur Data dan Algoritma

Perancangan Struktur Data

1. Untuk menyimpan sebuah elemen matriks, struktur data yang digunakan digambarkan di bawah ini,



dengan R = baris, C = kolom, V=nilai yang disimpan, NC=pointer ke kolom berikutnya, NR=pointer ke baris berikutnya.

Deklarasi programnya :

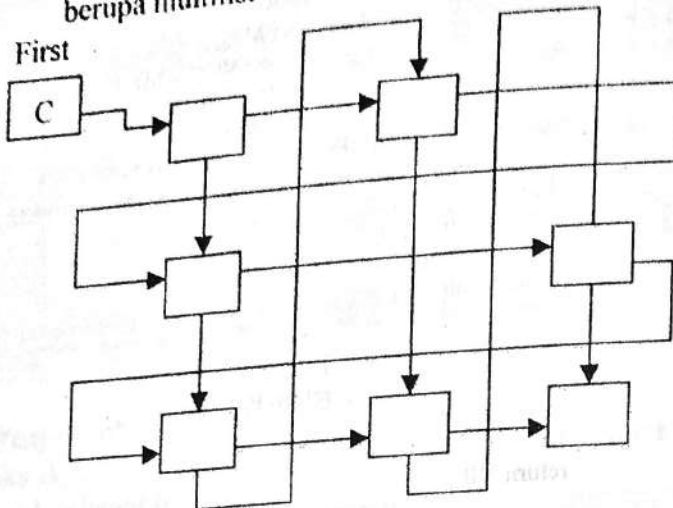
```

typedef struct TCell *PCell;
typedef struct TCell {
    PCell NextRow, NextCol; /* link ke elemen berikutnya */
    int Row, Col;           /* Baris dan Kolom */
    float Value;            /* nilai matriks */
}
  
```

TCell:

2. Untuk menyimpan sebuah matriks adalah berupa multilist

First



```
typedef struct TMatrix *PMatrix;
typedef struct TMatrix {
    PCell First; /* link ke cell [1,1] */
    int NRow, NCol; /* Dimensi matrix */
    int CRow, CCol; /* Current Row, Col */
    PCell CC, PC; /* Couple Current/Prev Cell */
    int Error; /* counter error matriks, 0 berarti oke */
} TMatrix;
```

Keterangan :

TCell = objek satu elemen matrix

TMatrix = objek matrix, berupa multi list TCell

First : menunjuk ke Cell[1,1] pada matrix

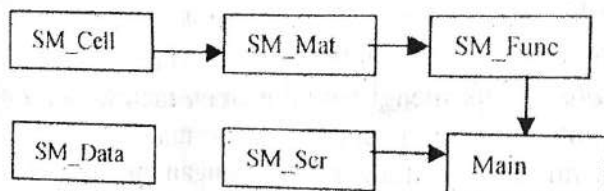
Error : menunjukkan banyaknya kesalahan isi matriks biasanya terjadi jika gagal menginsert Cell

CCol, CRow : current Row dan Col berisi [0,0] bila Current tidak valid (End)

CC : Current Cell. Menunjuk Cell M[CCol, CRow] NIL bila Cell M[CCol, CRow] tidak ada (=0)

PC : Previous Cell. Menyimpan CC terakhir yang tidak NIL, NIL bila CC = Cell M[1,1] (First)

Perancangan modul program sebagai berikut :



Penjelasan masing-masing modul :

SM_Cell : untuk mendefinisikan sel (elemen) matriks serta alokasi dandecalokasi memori

SM_Mat: untuk mendefinisikan *sparse* matriks serta operasi-operasi primitif yang dimilikinya

SM_Func : untuk mendefinisikan fungsi-fungsi penjumlahan, pengurangan dan perkalian matriks

SM_Data: untuk mendefinisikan data global *sparse* matriks

SM_Scr: untuk mendefinisikan antar muka program *sparse* matriks
SM_Main: program utama *sparse* matriks
Beberapa potongan algoritma dari modul yang ada sebagai berikut :

Modul SM_Cell :

```
void CAllocate(PCell *PC, int I, int J);
/* mengalokasi Cell baru dan mengassign baris, kolom
IS : - FS : PC = alokasi baru, Row(PC) = I, Col(PC) = J
PC = NIL jika alokasi gagal */
{
    *PC = (PCell) malloc(sizeof(TCell));
    if (PC != NIL) {
        Row(*PC) = I;
        Col(*PC) = J;
        Value(*PC) = 0;
        NextRow(*PC) = NIL;
        NextCol(*PC) = NIL;
    }
}

void CDeallocate(PCell *PC);
/* mendecalokasi Cell
IS : PC valid, FS : memori PC didekalokasi, PC = NIL */
{
    free(*PC);
    *PC = NIL;
}
```

Modul SM_Mat :

```
void MSearchRow(PMatrix PM, int I, int J, PCell *RPrev, PCell *PC)
/* Mencari Cell PM(I,J) dengan traversal row
* Jika ketemu : PC = Cell PM(I,J), RPrev = cell baris sebelum PC
* Jika tidak : PC = NIL, RPrev = cell baris terakhir sebelum posisi I,J
*/
{
    boolean found;
    *RPrev = NIL;
    *PC = First(PM);
    found = FALSE;
    /* tidak ada kasus kosong */
    do {
        if ((Row(*PC) == I) && (Col(*PC) == J)) {
            found = TRUE;
        }
        else if ((Col(*PC) > J) || ((Col(*PC) == J) && Row(*PC) > I)) {
            *PC = NIL;
        }
        else {
            *RPrev = *PC;
            *PC = NextRow(*PC);
        }
    } while (*PC != NIL && (!found));
}

/****** TRAVERSAL Current Cell *****/
void MGoFirst(PMatrix PM)
{
    CRow(PM) = 1;
    CCol(PM) = 1;
    PM->CC = First(PM);
    PM->PC = NIL;
}
```



```

void MtGoRow(PMatrix PM, int I, int J)
{
    MtSearchRow(PM, I, J, &PM->PC, &PM->CC);
    CRow(PM) = I;
    CCol(PM) = J;
}

void MtNextCol(PMatrix PM)
{
    CCol(PM) = CCol(PM) + 1;
    if (CCol(PM) > NCol(PM)) {
        CRow(PM) = CRow(PM) + 1;
        if (CRow(PM) > NRow(PM)) {
            CRow(PM) = 0;
            CCol(PM) = 0;
            PM->CC = NIL;
            PM->PC = NIL;
            return;
        }
        CCol(PM) = 1;
    }
    if (PM->CC != NIL)
        PM->PC = PM->CC;
    PM->CC = NextCol(PM->PC);
    if (PM->CC != NIL) {
        if ((Row(PM->CC) != CRow(PM)) || (Col(PM->CC) != CCol(PM))) PM->CC = NIL;
    }
}

```

Modul SM Func :

```

PMatrix MtAdd(PMatrix PM1, PMatrix PM2)
{ PMatrix PM;

    if ((NRow(PM1) == NRow(PM2)) && (NCol(PM1) == NCol(PM2)))
    {
        MtAllocate(&PM, NRow(PM2), NCol(PM1));
        if (! MtsEmpty(PM)) {
            MtGoFirst(PM);
            MtGoFirst(PM1);
            MtGoFirst(PM2);
            while (! MtsEnd(PM1)) {
                MtSet(PM, MtGet(PM1) + MtGet(PM2));
                MtNextRow(PM);
                MtNextRow(PM1);
                MtNextRow(PM2);
            }
            return PM;
        }
        return NIL;
    }
}

PMatrix MtMul(PMatrix PM1, PMatrix PM2)
{
    * Menghitung PM1*PM2
    * IS : PM1, PM2 valid
    * FS : return PM1*PM2
    * return NIL jika :
    * matrix tidak bisa dikalikan (dimensi beda)
    * hasil tidak berhasil dicreate (memory tidak cukup)
    PMatrix PM;
    float SumMt;
    int i, j, k;
    if (NCol(PM1) == NRow(PM2)) {
        MtAllocate(&PM, NRow(PM1), NCol(PM2));
    }
}

```

```

if (! MtsEmpty(PM)) {
    MtGoFirst(PM);
    for (i = 1; i <= NRow(PM); i++) {
        MtGoFirst(PM2);
        for (j = 1; j <= NCol(PM); j++) {
            SumMt = 0.0;
            MtGoCol(PM1, i, 1);
            for (k = 1; k <= NRow(PM2); k++) {
                SumMt = SumMt + MtGet(PM1) * MtGet(PM2);
            }
            MtNextRow(PM2);
            MtNextCol(PM1);
            MtSet(PM, SumMt);
            MtNextCol(PM);
        }
        return PM;
    }
    return NIL;
}

```

3. Hasil Aplikasi Program

Berikut diberikan beberapa contoh *snapshot* program yang telah diuji kebenarannya.

3.1. Tampilan awal



Tampilan di atas merupakan tampilan awal program *Matlator v1.1*, berupa fitur *Select* untuk memilih matriks yang akan diakses (A sampai Z), *New* untuk menentukan orde matriks yang baru, *Delete* untuk menghapus matriks, *Move* untuk memilih baris atau kolom yang akan diakses, *Value* untuk mengisi nilai elemen matriks serta *Function* untuk memilih operasi pada matriks. Pada tampilan awal di atas diinisialisasi dengan mengakses matriks A dengan status *empty* artinya listnya masih kosong (nil).

3.2. Tampilan Sparse Matriks

Sewaktu orde sudah didefinisikan maka tampilan berupa *sparse* matriks dengan semua elemen diset nol, dan posisi kursor (*bold*) menunjukkan elemen yang sedang di akses. Terlihat pada gambar 1.

Gambar 1.

3.3. Penjumlahan matriks $C = A + B$

Matriks $A_{20 \times 11}$, dengan **bold kuning** menyatakan elemen tidak nol sebagai berikut :

Matriks $B_{20 \times 11}$, dengan **bold kuning** menyatakan elemen tidak nol sebagai berikut :

Matriks $C_{20 \times 11}$ hasil dari $A + B$ dengan **bold kuning** menyatakan elemen tidak nol sebagai berikut :

3.4. Perkalian matriks $E = C \times D$

Matriks $D_{11 \times 10}$, dengan **bold kuning** menyatakan elemen tidak nol sebagai berikut :

Matriks $E_{20 \times 10}$ hasil dari $C \times D$ dengan **bold kuning** menyatakan elemen tidak nol sebagai berikut :

4. Kesimpulan

Struktur data yang paling cocok untuk mengelola *sparse* matriks adalah dengan multilist. Dalam penelitian ini telah berhasil dikembangkan program operasi dasar penambahan, pengurangan serta perkalian pada *sparse* matriks dengan menggunakan struktur data multilist representasi pointer. Efisiensi memori yang diperoleh harus dibayar mahal dengan sulitnya algoritma yang harus dikembangkan.

Referensi

- [1] Doerr, Alan, *Applied Discrete Structures for Computer Science*, Science Research Associates, 1985
- [2] Horowitz, Ellis, *Fundamentals of Data Structures*, Galgotia, 1999
- [3] Inggriani Liem, Diktat Kuliah Struktur Data dan Algoritma Pemrograman, Jurusan Teknik Informatika, ITB, 1996
- [4] Kernighan, Brian W, *The C Programming Language*, Prentice Hall, 1998